(12) UK Patent Application (19) GB (11) 2 233 480 (13) A

(43) Date of A publication 09.01.1991

(21) Application No 9006508.7

(22) Date of filing 23.03.1990

(30) Priority data
(31) 8914352 (32) 22.06.1989 (33) GB

(71) Applicant
International Computers Limited

(Incorporated in the United Kingdom)

ICL House, Putney, London SW15 1SW, United Kingdom

(72) Inventors
Michael William Berry Curran
Richard Norcott Taylor

(74) Agent and/or Address for Service
D C Guyatt
STC Patents, West Road, Harlow, Essex, CM20 2SH, United Kingdom

(51) INT CL[5]
G06F 13/368 15/16

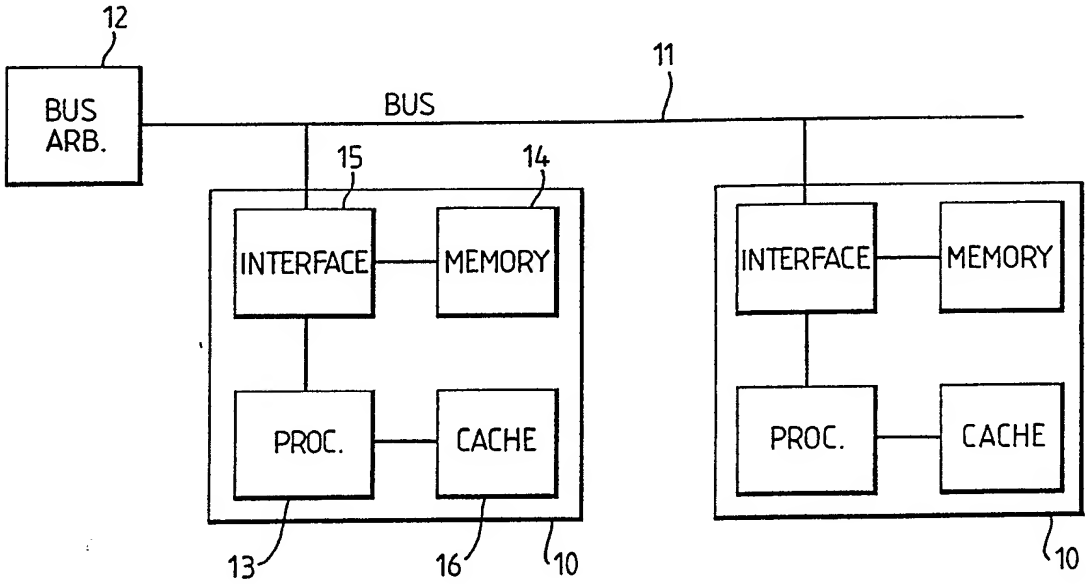
(52) UK CL (Edition K)
G4A AMP

(56) Documents cited
EP 0320607 A2

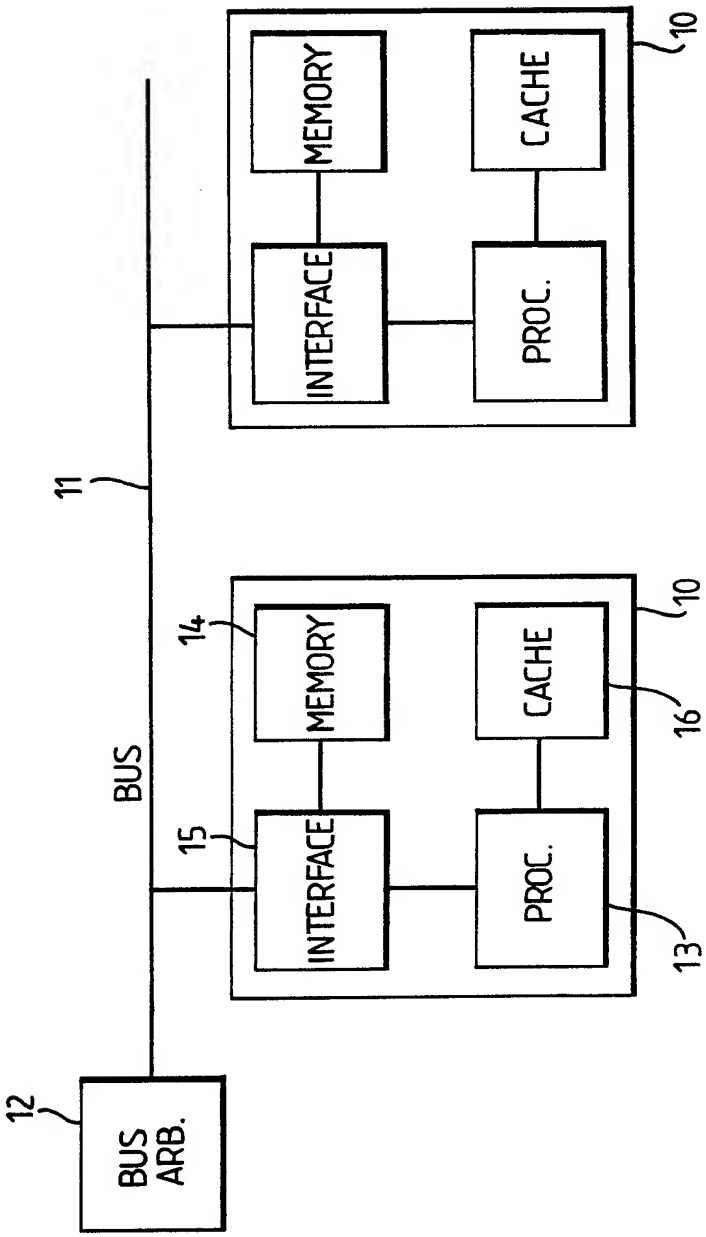
(58) Field of search
UK CL (Edition K) **G4A AFGDR AFGL AMP**
INT CL[5] **G06F**
On-line databases: WPI; INSPEC

(54) Multiprocessor data processing system

(57) A multi-processor data processing system is described, in which each processor has a local memory. The local memories together from the main memory of the system, and any processor can access any memory, whether it is local to that processor, or remote. Each processor has an interface circuit which determines whether a memory access request relates to the local memory or to a remote memory, and routes the request to the appropriate memory, remote requests being routed over a bus. Whenever as write access is made to the local memory, a dummy write request is routed over the bus to all the other processors. Each processor monitors all write requests on the bus, and if a copy of the location specified in the request is held in a local cache memory, that copy is invalidated, so as to ensure cache consistency.



GB 2 233 480 A


BUS ARB.
12
BUS
11
INTERFACE
PROC.
MEMORY
CACHE
10
15
14
16
13

Multiprocessor data processing system

Background to the invention

This invention relates to multiprocessor data processing systems.

In a conventional multiprocessor system, a number of processors share a common memory, which they access over a shared memory bus. Whenever a processor requires to access data in the memory, it first has to win control of the bus, and then places a memory access request on the bus. In the case of a write request, the processor also sends the data to be written. In the case of a read request, the memory responds by returning the required data to the processor.

A problem with such an arrangement is that the memory bus presents a bottleneck which severely restricts the operation of the system. This problem becomes more acute as the number of processor in the system increases.

"Multi-Microprocessors: an overview and working example", by S.M. Fuller et al, Proceedings of the IEEE, February 1978, page 216, describes a possible solution to this problem. A multiprocessor system is described, comprising a number of computer modules (Cm), each of which consists of a processor, local memory, input/output devices, and a local switch (Slocal). A processor can directly reference any location in any of the memories/ The local switch decides whether the location is in the local memory of the module and, if so, passes the reference directly to the local memory.

If, on the other hand, the referenced location is in some other module, the local switch passes the reference over a bus to a mapping controller (Kmap). The mapping controller then initiates a memory request to the module which contains the referenced location.

It can be seen that this reduces the amount of traffic over the bus, since references to data held in the local memory can be dealt with without any transfers over the bus. However, a problem with this is that it requires a special mapping controller external to the individual processing modules. As a result, in moving from a single-processor system to a multi-processor system, not only is it necessary to add extra processing modules, but it is also necessary to introduce a mapping controller. Moreover, there has to be a special memory management activity to configure the mapping controller, in addition to that required to configure the local switches in the processing modules.

The object of the present invention is to overcome these problems.

Summary of the invention

According to the invention there is provided a data processing system comprising, a plurality of processing modules interconnected by a bus, each module comprising a processor, a memory, and an interface unit, wherein whenever a processor requests a memory access, the interface unit in the same module determines whether the requested memory location is in the memory in the same module and, if so, directly access that location, whereas, if the requested memory location is in the memory of a remote module, the interface unit sends a memory access request direct to that remote module.

Brief description of the drawings

The sole figure of the drawing is a block diagram of a multi-processor data processing system embodying the present invention.

Description of an embodiment of the invention

One data processing system in accordance with the invention will now be described by way of example with reference to the accompanying drawing.

Referring to the drawing, the system comprises a plurality of data processing modules 10, interconnected by a bus 11. Only two modules are shown in this example, but it will be appreciated that the system can be expanded by connecting further modules to the bus. Accesses to the bus are controlled by a bus arbitration circuit 12.

Each of the processing modules 10 comprises a data processor 13, a memory 14, an interface circuit 15 and a cache memory 16.

The memories 14 in the modules 10 together form the main memory of the system. Any processor can access any location in any of the memories 14, whether it is local to that processor (i.e. in the same modules) or remote (i.e. in a different module).

The cache memories 16 are limited capacity very high performance data storage units, associated with the individual processors. Each cache is dynamically mapped on to the individual locations of the main memory, so as to provide rapid access without the delays involved in accessing the slower main memory. Cache memories are well known as such, and so it is not necessary to describe the structure of the caches 16 in any further detail.

When any of the processors 13 requires to access a memory location, it issues a read or write access request to the interface circuit 15. The interface circuit determines whether the requested memory location is in the local memory 14 in the same module, or is in a remote module. This is achieved by consulting page tables which indicate for each page whether that page is local or remote. These tables may be the same as those conventionally provided for address

translation, and hence do not impose any additional initialisation overhead on the system.

If the requested location is in the local memory, the interface unit accesses this location directly, and reads or writes the data as required.

If the requested location is remote, the interface circuit sends a request signal to the bus arbitration circuit 12, requesting permission to use the bus. The arbitration circuit ensures that only one of the interface circuits has access to the bus at any given time. When the interface circuit has been granted permission to use the bus, it sends the memory access request over the bus to the remote module that contains the requested memory location. This request is received by the interface circuit in that remote module, which then accesses the memory 14 in that module so as to read or write the data as required. In the case of a read request, the data is then returned over the bus to the requesting module.

In the case of a local write request, as well as writing the data to the local memory, the interface circuit also sends a dummy write request on the bus. The purpose of this will be explained later.

The operation of the cache memory will now be described.

Whenever a processor 13 issues a memory access request, the cache checks whether a copy of the requested memory location is already present in the cache. If so, the data can be accessed very rapidly. In the case of a read access, no main store access is necessary. However, in the case of a write access, the data is also written back to the main memory, so as to ensure that the main memory is kept up to date.

If, on the other hand, the required data is not present in the cache, it is accessed from the main memory as described above, and copied into the cache, so that it is available for future access.

It is possible that several copies of the same memory location may exist in the caches of different modules 10, and these copies must be kept consistent. In the present system, this is achieved as follows.

As described above, whenever any processor writes to any memory 14, a write request is sent over the bus 11. In the case of a local write, this involves a dummy write request over the bus. Each processing module monitors the bus, and whenever it detects a write request from another module, it checks whether a copy of the data in question is held in its own cache 16. If so, the cache copy is invalidated. This ensures that different versions of the same data cannot exist in different caches.

Conclusion

In summary, it can be seen that a local read access request does not involve the use of the bus 11, since the data can be read directly from the local memory 14. This significantly reduces the amount of traffic on the bus, and hence reduces the bottleneck problem.

The system does not require any external unit for routing remote access requests between different processing modules, since this is performed by the interface circuits 15 in the modules.

CLAIMS

1. A data processing system comprising, a plurality of processing modules interconnected by a bus, each module comprising a processor, a memory, and an interface unit, wherein whenever a processor requests a memory access, the interface unit in the same module determines whether the requested memory location is in the memory in the same moduls and, if so, directly access that location, whereas, if the requested memory location is in the memory of a remote module, the interface unit sends a memory access request direct to that remote module.

2. A system according to Claim 1 wherein each module also includes a cache for holding copies of data locations in the memories.

3. A system according to Claim 2 wherein whenever an interface unit performs a write access to the memory in the same module, it also sends a dummy write access request over the bus to all the other modules, and wherein each module is arranged to monitor all write access requests on the bus from other modules, including dummy write access requests, to check whether a copy of the location specified by that request is present in the cache memory of that module and, if so, to invalidate that copy.

4. A data processing system substantially as hereinbefore described with reference to the accompanying drawing.

Published 1991 at The Patent Office, State House, 66/71 High Holborn, London WC1R 4TP. Further copies may be obtained from The Patent Office. Sales Branch, St Mary Cray, Orpington, Kent BR5 3RD. Printed by Multiplex techniques ltd, St Mary Cray, Kent, Con. 1/87